Graph model based Trajectory Prediction for traffic agents

Arjun Srinivasan Ambalam, Arun Kumar Dhandayuthabani, Rajeshwar N S, Sahana Anbazhagan
University of Maryland, College Park, USA

***Abstract*— The trajectory prediction problem is one where we estimate the future positions of all objects in a scene based on their trajectory histories. Trajectory forecasting for road agents in a heterogeneous traffic environment using a deep learning based approach. Accurate trajectory prediction is crucial for safe navigation. We try to account for the interactions between road agents by representing them as nodes of a graph. This is a method that uses graphs to represent the interactions between close objects. Previously, there were novel schemes proposed to predict the trajectories for traffic agents around a self-driving car efficiently such as GRIP [1] and GRIP++ [2]. We propose certain modifications in the network such as social pooling and using LSTM instead of GRU to obtain better prediction accuracy.**

I. Introduction

Autonomous driving has gone from "might be conceivable" to "commercially feasible" over the years, and credits to the innovations in the fields of computer vision, sensor signal processing and hardware design among the other. However, the two traffic accidents caused by self-driving cars by Tesla and Uber in 2018 stirred a lot of anxiety among the buyers regarding the safety of the autonomous cars. Therefore, it is critical to strengthen the performance of the intelligent algorithms running on autonomous driving cars, and prediction of the future trajectories of neighboring objects is one such intelligent algorithm. Experts contend that we can avoid such a traffic accident if each autonomous driving car involved could precisely predict the locations of its surrounding objects in its environment.

Driving in an urban environment is much more challenging than driving on a highway. Urban traffic is riddled with more uncertainties, complex road conditions, and diverse traffic-agents, especially on some cross-roads..Some of the challenges arise in dense urban environments, where the traffic consists of different kinds of traffic-agents, including cars, bicycles, buses,pedestrians, etc.. These traffic-agents have different shapes,dynamics, and varying behaviors and can be regarded as an instance of a heterogeneous multi-agent system. To guarantee the safety of autonomous driving, the system should be able to analyze the motion patterns of other traffic-agents and predict their future trajectories so that the autonomous vehicle can make appropriate navigation decisions.

However, we must regard the fact that it is very challenging to predict the motion surrounding objects, given that there are so many factors which affect the future trajectory of am object. Earlier methods suggest that in order to predict the future locations by recognizing maneuvers. But, this is subject to the fact that the method fails to predict the position of the objects accurately when the type of the maneuver is recognized incorrectly. This could possibly happen if a particular scheme predicts only based on the sensors like GPS and miss out on the visual clues. e.g., turn signals. This generally happens when the prediction is based purely on the sensors like GPS and misses out on other visual information.

Fig. 1: The problem is prediction of short-term (1–3 seconds) and long-term (3-5 seconds) spatial coordinates of various road-agents such as cars, buses, pedestrians, etc.. by observing their current and previous positions. The red curve represents the predicted trajectory of the vehicles.

Karasev et al. [3] proposed that the motion of pedestrians could be predicted by modeling their behaviours as Markov jump processes. But, though they claimed that the route of an observed pedestrian could be predicted, their method required a semantic map. Apart from this another problem was that they needed one or several goals of the pedestrians, which is not very good in case of self-driving cars. This is because an autonomous car cannot have the knowledge about the pedestrian's destination in advance. Bhattacharyya et al. [4] predicted the future vehicle odometry sequence by trying to predict the bounding boxes of objects in RGB camera frames. But still, the predicted bounding boxes were required to be mapped to the coordinate system of the autonomous car. If not, the self-driving car cannot make a correct response to these predicted locations.

Besides, almost all schemes consider only the state of one predicted object and do not consider the state of the surrounding objects. But, the states of the surrounding objects are crucial for motion prediction. In order to have a safe operation of the autonomous vehicles, their navigation and perception systems have to be able to predict their future locations after analyzing the motion patterns of the surrounding traffic agents so that the vehicles can make sound driving decisions. We have used a robust and efficient object trajectory prediction scheme called GRIP, which can infer future locations of surrounding objects simultaneously and is trainable end-to-end.

The rest of the paper is organized as follows. We give a brief overview of related prior work in Section 2. In Section 3, we define the problem and give details of our prediction

algorithm. We test out our algorithm on the Apolloscape data set and show the performance of our methods in Section 4.

II. Related Works

A. Classical Methods

Here, we have a brief outline of some classical techniques. Trajectory prediction is a topic that has been extensively researched about. The various approaches explored include the Bayesian formulation, the Monte Carlo simulation [5] [6], Hidden Markov Models (HMMs) [5] [7], and Kalman Filters [8] [9]. If the method modeled the road agent interaction then they were regarded as more accurate, and if they were unable to do so they were regarded to be sub-optimal and less accurate methods [10]. Most of these models accounted for the pedestrian interaction in crowds and thus helped in improving the accuracy [11]. Various velocity obstacles based techniques have been extended using kinematic constraints for modelling the heterogeneous road agents interactions [12].

B. Deep-Learning based Methods

There are quite a few Deep-Learning methods that have emerged to tackle this problem nowadays. We are discussing a few of them here. TraPHic [5] is a new algorithm for predicting the near -term trajectories of road agents in dense traffic scenarios. This approach is designed for heterogeneous traffic conditions where the roads have many agents like busses, cars, motorbikes, bikes, or pedestrians. the interactions between different road agents are modeled using a novel LSTM-CNN hybrid network for trajectory prediction. The heterogeneous interactions implicitly account for the changing shapes, dynamics, and behaviors of different road agents. These factors are specifically accounted for and the horizon-based interactions are modeled to implicitly model the driving behavior of each road agent. The performance of TraPHic is evaluated on the standard datasets, then a new dense, heterogeneous traffic dataset corresponding to urban Asian videos and agent trajectories. This method outperforms the state-of-the-art methods by 30% for the dense traffic datasets [5]. There are approaches based on deep neural networks which use variants of Recurrent Neural Networks (RNNs) for sequence modeling.

C. Hybrid Models

Deep-learning-based hybrid methods consist of networks that integrate two or more deep learning architectures and a few examples of this includes CNNs, GANs, VAEs and LSTMs. These have been extended to hybrid networks by combining RNNs with other deep learning architectures for motion prediction. They are natural generalizations of feedforward neural networks to sequence, and the advantages of RNNs makes it an apt choice for traffic prediction. But since RNNs cannot be used for modeling long-term sequences, so many traffic trajectory prediction methods use long short-term memory networks to model road-agent interactions. These can also be used for trajectory prediction of pedestrians in crowd as well. There is quite a lot of work going on towards the development of hybrid networks. Generative models are successfully being used for tasks such as super resolution, image-to-image translation, and image synthesis. Generative models such as VAEs and GANs have been used for trajectory prediction of pedestrians in a crowd and in sparse traffic although their application in trajectory prediction is limited due to back-propagation during training is non-trivial.

Yet another relatively new approach in this field is Trajectron++ [13], which is a modular, graph-structured recurrent model. This shows the trajectories of a general number of diverse agents while incorporating agent dynamics and heterogeneous data. It is designed to be closely connected with robotic planning and control frameworks.

D. Deterministic, Generative and Probabilistic Approaches

The earliest works in the field of human trajectory prediction has been based on deterministic models. Most approaches have been applied to the issue of trajectory forecasting, they were formulated into a time-series regression problem and then methods like Gaussian Process Regression (GPR), Inverse Reinforcement Learning (IRL) [14], and Recurrent Neural Networks (RNN) [15] [16] were applied to good effect. IRL is based on the assumption of the interaction outcome, making modeling multimodal data difficult. Since robotic use cases require fast inferences GPR fails in such cases since it has long inference times.

Generative approaches gained attention off late, in the field of trajectory forecasting due to the advancements in deep generative models. GAN-based models can directly produce position distributions and CVAE-based recurrent models usually rely on a bivariate Gaussian Mixture Model (GMM) to output position distributions. Both these output structures make it difficult to enforce dynamic constraints such as those arising from no side-slip conditions, for example non-holonomic constraints. At present, the deterministic and generative fields are dissimilar in that models are exclusively designed to either produce one trajectory or a distribution of trajectories. While probabilistic models may produce these alone, there arises questions regarding multimodal distributions and whether the mean corresponds to feasible trajectories. Umpteen number of recent methods are GAN-based and thus unable to obtain any distributional information beyond sampling.

III. Approach

A. Problem Formulation

Our approach is to generate reasonable number of trajectory distributions for varying number of road agents present in the scene. The model output contains future (x,y) coordinates of all the agents over the prediction horizon.

$$Y = [p^{t_h+1}, p^{t_h+2}, ..., p^{t_h+t_f}] \tag{1}$$

where $p^{(t)} = [x_0^{(t)}, y_0^{(t)}, ..x_n^{(t)}, y_n(t)]$ containing the x,y coordinates of the road over the predicted time horizon t_f.

B. Scene Representation

A spatiotemporal graph in [1] $G = (V,E)$ is used to represent the inter-object interaction which resembles a social network where $V = \{v_{it} | i = 1,...,n, t = 1,...,t_h\}$ is the node set containing the coordinates of n road agents at time t. E is the edge set containing spatial edge information and inter-frame information. The spatial information [2] is represented by the $l2$ distance between two objects. The inter frame edges represents the historical information in temporal space denoted as $E = v_{it} v_{i(t+1)}$ which represents a trajectory of an object over its time steps. This relationship is represented by an adjacency matrix $A = \{A_0, A_1\}$ where A_0 is the Identity matrix I and A_1 is a spatial connection matrix.

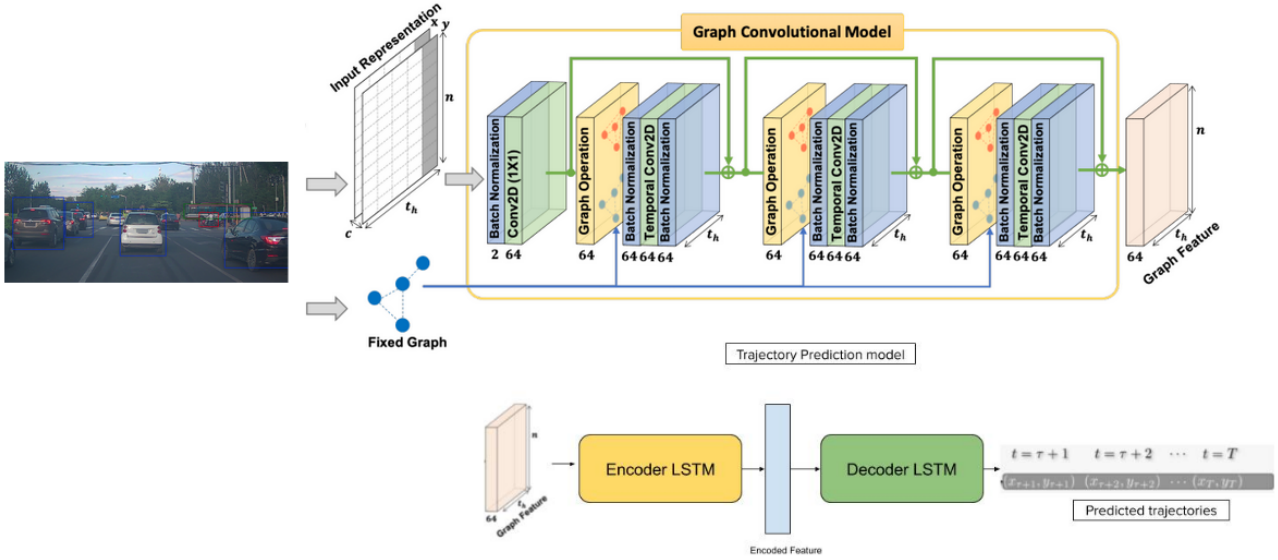

Fig. 2: *Our modified architecture*

C. Graph Model

The input to the model is represented as $(n \times t_h \times c)$ where n are the number of road agents,t_h are the time steps and c are the number of coordinates representation which $c = 2$ in this case. The input representation is fed to a 2D Convolutional $(1x1)$ layer to increase the number of channels. Unlike the single Conv2D in [1], multiple conv2D's were also stacked to represent the (x, y) to higher dimensional space for better data representation. This is fed to several graph operations which describes the relationship between various agents and temporal convolutions are used to capture the movement patters.

1) Graph Convolutional layer: As mentioned earlier, we try to borrow techniques from both GRIP and the Enhanced version GRIP++ by using both Fixed and Dynamic graphs for representing inter-agents relationship [1]. The output of the Graph Convolution layer is calculated as

$$f_{graph} = \sum_{j=0}^{1} (G^j_{fixed} + G^j_{train}) f_{conv} \tag{2}$$

where G_{fixed} and G_{train} are the fixed and trainable graphs respectively

2) Temporal Conv layer: We input the generated feature fgraph to a Temporal Convolutional layer.And set appropriate paddings and strides to make sure that each layer has an output feature map with the expected size.

3) Trajectory Prediction Model: This model predicts the future trajectories for all observed objects in a scene. The Trajectory Prediction Model is an LSTM encoder-decoder network that takes the computed output of the Graph Convolutional Model graph's input.The output of the graph convolutional model is fed into the encoder LSTM at each time step. Then, the hidden feature of the encoder LSTM, as well as coordinates of objects at the previous time step, are fed into a decoder LSTM to predict the position coordinates at the current time step. We get the predicted results from the networks, and average the results (predicted velocities) at each time step. After getting the averaged predicted velocities, we add them (Δx ,Δy) back to the last historical location to convert the predicted results to (x, y) coordinates.

D. Training and Optimization

The model is implemented using Pytorch Library in Python Language. As we are continuously predicting the coordinates, we model this as a regression problem in which the loss can be computed as

$$Loss = \frac{1}{t_f} \sum_{t=1}^{t_f} \|Y^t_{pred} - Y^t_{GT}\| \tag{3}$$

We use the Adam Optimizer with batch sizes 64/128 and trained the model for 50 epochs.

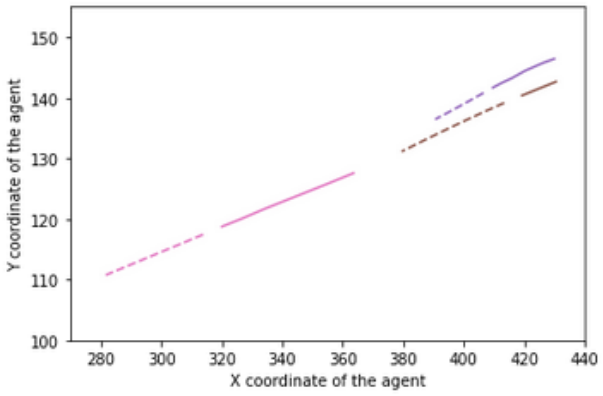

Fig. 3: *We tried to visualise our prediction results here. The agents are represented in different colors, dashed line represents our predicted trajectory and solid line represents the current and previous frames from ground truth.*

IV. Experiments Results

We ran the network on a Ubuntu 18.04 Desktop with Intel i7 CPU,16 GB RAM and a RTX 2060 GPU.

A. Dataset

We use the ApolloScape Trajectory dataset which is acquired by Baidu by running their cars in urban areas during heavy traffic scenarios. Both Camera Images and LIDAR point clouds are collected using Object detection and tracking algorithms. There are 53 minutes training sequences and 50 min training sequences each at 2 fps. Each line in the file contains frame_id, object_id, object_type, position_x, position_y, position_z, object_length, object_width, object_height, heading. There are five different types of objects namely small vehicles, big vehicles,pedestrians,motorcyclist and bicyclist and others based on which unique Object IDs are assigned. Thus it consists of a highly complicated traffic flows mixed with vehicles, riders and pedestrians. The positions are given in the world coordinate systems in meters. We chose 80% sequences from training set and remaining 20% as validation sequences.

B. Pre-processing

We take the traffic scene within 180 feet (± 90 feet). All objects within this region will be observed and predicted in the future. While constructing the graph, we consider two objects are close if their distance is less than 25 feet (Dclose = 25). For better generalization capability, we applied data augmentation on the input data, e.g., randomly rotate the input data or enhance the model using testing observed data.

C. Metrics

ApolloScape uses the Average displacement error (ADE) and Final Displacement Error (FDE) to measure the performance of the models. ADE - This indicates the mean Euclidean distance over all the predicted positions and ground truths during the prediction time FDE - This describes the mean Euclidian distance between the final predicted positions and the corresponding ground truths. Since the trajectories of cars,bicyclist and pedestrians have different scales, we use the weighted sum of ADE (WADE) and weighted sum of FDE (WFDE) as metrics.

$$WSADE = D_v.ADE_v + D_p.ADE_p + D_b.ADE_b \quad (4)$$

$$WSFDE = D_v.FDE_v + D_p.FDE_p + D_b.FDE_b \quad (5)$$

where D_v,D_p,D_b are related to reciprocals of the average velocities of vehicles,pedestrian and cyclist in the dataset. We adopt 0.20,0.58,0.22 respectively.

D. Model Study

We used LSTM instead of GRU in GRIP++ and observe that performance up gradation is not much and GRU's perform slightly better than LSTM. This may be due to fact that Appolloscape dataset contains smaller amount of training samples. LSTM generally needs huge amount of data to train properly.

V. Conclusion

We were able to implement existing models used for trajectory prediction.Also,we were able to study the network architectures and add some modifications to study the effects.Some of the works that we would like to focus on are using traffic psychology to do behavior classification of road agents and produce better trajectory prediction results. The dynamics of the vehicle or object type is not being considered explicitly in the present work, these external details could have added more value to model the interaction among other heterogeneous agents.

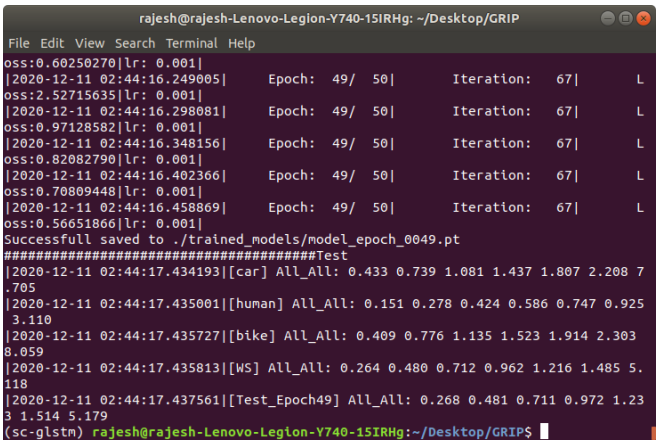

Fig. 4: *Terminal displaying trained model results*

References

[1] X. Li, X. Ying, and M. C. Chuah, "Grip: Graph-based interaction-aware trajectory prediction," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3960–3966.

[2] ——, "Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving," *arXiv preprint arXiv:1907.07792*, 2020.

[3] V. Karasev, A. Ayvaci, B. Heisele, and S. Soatto, "Intent-aware longterm prediction of pedestrian motion," *IEEE International Conference on Robotics and Automation (ICRA) pp.2543–2549*, 2016.

[4] A. Bhattacharyya, M. Fritz, and B. Schiele, "Long-term on-board prediction of people in traffic scenes under uncertainty," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition pp. 4194–4202*, 2018.

[5] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, "Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8483–8492.

[6] S. Lefevre, C. Laugier, and J. Ibanez-Gusman, "Exploiting map information for driver intention estimation at road intersections." IEEE, 2011.

[7] S. Danielsson, L. Petersson, and A. Eide-hall, "Monte carlo based threat assessment: Analysis and improvements." IEEE, 2007.

[8] A. Bera, S. Kim, T. Randhavane, S. Pratapa, and D. Manocha, "Glmp-realtime pedestrian path prediction using global and local movement patterns," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 5528–5535.

[9] J. Firl, H. Stubing, S. A. Huss, and C. Stiller, "Predictive maneuver evaluatoin for enhancement of car-to-x mobility data." IEEE, 2012.

[10] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference*, p. 797–803, 2010.

[11] A. Bera, S. Kim, T. Randhavane, S. Pratapa, and D. Manocha, "Glmp-realtime pedestrian path prediction using global and local movement patterns," *Robotics and Automation (ICRA), IEEE International Conference*, p. 5528–5535, 2016.

[12] Y. Ma, D. Manocha, and W. Wang, "Autorvo: Local navigation with dynamic constraints in dense heterogeneous traffic," *Computer Science in Cars Symposium (CSCS), ACM*, 2018.

[13] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control," *arXiv preprint arXiv:2001.03093*, 2020.

[14] N. Lee and K. Kitani, "Predicting wide receiver trajectories in american football," *IEEE Winter Conf. on Applications of Computer Vision*, 2016.

[15] J. Morton, T. Wheeler, and M. Kochenderfer, "Analysis of recurrent neural networks for probabilistic modeling of driver behavior," *IEEE Transactions on Pattern Analysis Machine Intelligence 18(5), 1289-1298*, 2017.

[16] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention in human crowds," *In: Proc. IEEE Conf. on Robotics and Automation*, 2018.